Martini, B., Gharbaoui, M., Fichera, S., Castoldi, P. (2017). *Network Orchestration in Reliable 5G/NFV/SDN Infrastructures*. Paper submitted in 2017 19th International Conference on Transparent Optical Networks (ICTON). Girona: IEEE.

DOI:10.1109/ICTON.2017.8024937

Network Orchestration in Reliable 5G/NFV/SDN Infrastructures

B. Martini†, M. Gharbaoui*, S. Fichera*, and P. Castoldi*

† National Inter-University Consortium for Telecommunications (CNIT), Pisa, Italy

Tel: +39 050 549 2245, Fax: +39 050 549 2250, e-mail: barbara.martini@cnit.it

** Scuola Superiore Sant'Anna, Pisa, Italy*

ABSTRACT

In this paper, we elaborate an SDN orchestration solution aiming at the dynamic adaptation of service chain paths thereby addressing high-availability requirements of 5G applications. We present an SDN orchestrator that periodically monitors the availability of the network and, if necessary, promptly adapts service chain paths to recover from congestion events and to preserve network QoS performance of service data flows. A set of performance results are finally presented.

Keywords: orchestration, SDN, NFV, service chain, 5G, software-defined infrastructure, Internet of things.

1. INTRODUCTION

With the advent of Network Function Virtualization (NFV) technologies [1] and with the recent developments in cloud computing paradigm (e.g., fog computing [2]), the 5G infrastructure will be characterized by a technological convergence between the computing (i.e., Cloud) and communication (i.e., Telco) systems. These emerging trend portends a 5G network architecture with service platforms deployed as micro-clouds at the Edge of 5G infrastructure and composed of generalized Virtual Functions (VFs) providing both applications (e.g., IoT gateways or service overlays) and network services (e.g., middlebox appliances) that can be dynamically composed in the process of end-to-end service delivery (i.e., dynamic service chaining) [3]. On the other hand, Software-Defined Networking (SDN) provides programming abstractions that can be effectively exploited for the dynamic enforcement and in-line steering of data flows across VFs taking part in the service chains [4]. The effectiveness of SDN/NFV to offer enhanced functionalities to the service provider networks is now tangible as both technologies have started being steadily investigated and experimentally demonstrated [5][6]. Several works in the literature show that the adoption of SDN/NFV, possibly in combination with an orchestration layer, provides increasing flexibility and scalability to dynamic service chaining [7][8][9]. However, a limited interest has been devoted to the techniques that can to ensure reliability and QoS performance along the path followed by data flows while traverse chained VFs (i.e., service chain paths) [6].

In this paper we present an SDN orchestrator aiming at achieving reliable and QoS-enabled service chain paths for 5G services. Previous works of authors addressed the architectural aspects of NVF/SDN infrastructures with orchestration capabilities aligned with Service-Oriented Architecture (SOA) principles [11] and in combination with context-awareness features [12][13]. In this work, we go through a multi-layer orchestration concept for SDN/NFV/5G infrastructures, i.e., *software-defined infrastructure*. Specifically, we present an SDN-based orchestration system acting as a traffic-engineered service chaining solution able to dynamically adapt end-to-end service data paths upon either congestion events at switches/links or SLA violations of data flows, in order to ensure the desired level of data delivery in spite of the concurrent usage of network resources from different services.

2. MULTI-LAYER ORCHESTRATION FOR SOFTWARE-DEFINED INFRASTRUCTURES

The network *softwarization* and the advent of fog computing envisage a 5G scenario featured by a technological convergence and infrastructure sharing between Cloud and Telco systems that will make the major impact at the Edge of current infrastructure. As shown in Fig. 1, Edge networks, besides feeding wired/wireless access networks and aggregating traffics from users, will also include distributed micro-clouds of generalized VFs running on servers deployed in small datacenters and providing either applications and network services according to the "as-a-service" paradigm. As result of virtualization properties, e.g., self-contained service abstraction, heterogeneous resource capabilities offered by VFs can be uniformly exposed as *service components* with APIs and dynamically composed and provisioned in the process of end-to-end service delivery. Indeed, an end-to-end service can be provisioned as a composition of VFs by chaining application (e.g., data analytics) as well as network (e.g., virtual middlebox) service components required to properly process the application data flow [4]. The composite set of VFs in the chain is executed in a "*slice*" which is made of a set of logical resources (i.e., VMs or Containers) interconnected by a logical sequence of Virtual Links composing the service chain path. Virtual Links can be dynamically established by exploiting programming abstractions offered by SDN for the in-line steering of data flows across VFs.

In the envisioned 5G/NFV/SDN scenario, the effective and reliable provisioning of 5G services should rely on a (set of) orchestration process(es) able to cope with the heterogeneity of underlying resource infrastructures and the dynamicity of the 5G services while addressing application QoS requirements. As shown in Fig. 2, the

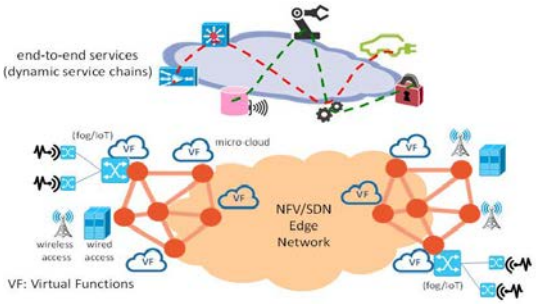

Figure 1. 5G network and service scenario.

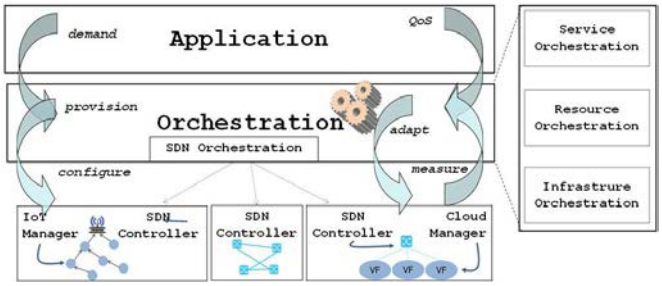

Figure 2. 5G functional layering.

orchestration process is in charge of the fulfilment of application demands (e.g., virtual network infrastructure for content delivery) by coordinately provisioning a composite set of *service components* (i.e., VFs) across different technological and/or administrative domains and exposing them as a single service instance. On the other hand, the orchestration is expected to guarantee the adequate service performance during the service lifecycle in spite of concurrent resource usage among users or service outages. To this purpose, the orchestration process also relies on monitoring functions to measure the status of the underlying (both physical and virtual) resources (e.g., load). Indeed, based on feedback from monitoring tools, the orchestration is expected to handle exceptions or deviations from normal workflows and to adapt provisioned resources to recover from service degradations or outages [14]. Given the heterogeneity of the infrastructure resources (i.e., cloud, network, IoT) possibly deployed in different provider domains and given the different functional areas involved (i.e., application, service and infrastructure), a stacked although interdependent set of orchestration layers are foreseen to address context-aware end-to-end service provisioning in 5G scenarios [4]. More specifically:

- *Service Orchestration Layer*

At this layer, the dynamic composition of *service components* is addressed according to a specified service graph specification, i.e., an ordered list of the required VFs types (e.g., firewall, deep packet inspection, data analytics), stating the invocation flow of *service components* as well as the Virtual Link requirements connecting VFs (e.g., maximum latency and/or minimum bandwidth) while not specifying yet the instances that should actually provide those VFs or the network data paths underpinning Virtual Links. This specification basically corresponds to the IETF Service Function Chain and to the ETSI Network Forwarding Graph concepts when also application *service components* are considered. This level of orchestration also deals with the adaptation of the service chaining logic to accommodate changeable user requirements or service contexts (e.g., changes in user location or preferences) or to recover from service degradation events (e.g., SLA violations) [4]. For instance, the service graph can be updated with the addition of *traffic acceleration* VF to address increased throughput needs of users. A Service-Oriented Architecture (SOA) approach can be used at this layer, especially to cope with multi-provider environments [11].

- *Resource Orchestration Layer*

At this layer, for a given chaining logic, the dynamic selection is carried out of (*i*) VF instances underpinning the required *service components* (i.e., VMs running specified software modules) and (*ii*) network domains and delivery path end points supporting connectivity between VFs (i.e., Virtual Links end points) to fulfill a specified request. Such (virtual) resources are selected among available candidates and their activation (i.e., service on-boarding) is triggered while leveraging lifecycle management systems or infrastructure orchestrators. Different algorithms can be used to perform selections while optimizing a specified utility function (e.g., minimization of the latency experienced by traffic flows). Such algorithms can consider the available capabilities in the clouds (e.g., processing capabilities at VF instances) and/or link capacities in the network domains (e.g., throughput at the Virtual Links) as well as their current load deduced from real-time monitoring data. Moreover, real-time monitoring data can be also used at runtime to adapt selections in case one or more VF instances are no more available (e.g., due to overload) or data delivery at Virtual Links degrades (e.g., throughput goes below a given threshold due to hotspots in the network domain). If such adaptation events are frequent, it means that the available resources (e.g., VF instances) are under-provisioned and scaling actions need to be carried out [11].

- *Infrastructure Orchestration Layer*

At this layer and at per-domain level, the delivery of VF services and Virtual Links is carried out through allocation of VMs into DC servers, service on-boarding into VMs (i.e., configurations and instantiation of VFs for them to promptly serve the end-users) and/or set-up of data delivery paths across a number of switches to connect VF instances. Moreover, the monitoring of VMs, servers and switches is performed in a way that if some hotspot occurs, different or augmented set of capabilities are activated (e.g., delivery path redirection throughout a different set of switches, VF scaling out). To this purpose, this layer leverages infrastructure managers related to IoT devices (e.g., ThingSpeak[15]), cloud platforms (e.g., OpenStack[16]) and SDN

networks (e.g., ONOS controller [17]) possibly enhanced with infrastructural abstractions and supporting intent-based invocations [18][10].

3. SDN ORCHESTRATION

SDN has been recently demonstrated also outside the traditional network domains, e.g. in SDN-based IoT and Cloud [19]. Hence, SDN can play a key role in harmonizing the control and the orchestration of network and data delivery services across IoT, Cloud/Fog and Edge network domains. In this paper we focus on the orchestration at the network infrastructure and resource level leveraging SDN across IoT, Cloud and Edge domains (i.e., SDN orchestration) aiming at properly assuring data delivery across service chain paths despite the concurrent usage of network capabilities from different services. Accordingly, we implemented an SDN orchestrator as an application running on top of the ONOS network controller platform in charge of offering control capabilities of the underlying network switching nodes through API [13]. This way, in principle the SDN orchestrator can flexibly operate over multiple ONOS controllers thereby addressing scalability. The SDN orchestrator exposes at NorthBound a RESTful interface which enables the upper layer applications [4][20] to directly demand *the set-up/tear-down of* service chain paths *by specifying an ordered set of end-point IP addresses of VF instances according to* an intent-based approach [13]. The provisioning details are then handled by the SDN orchestrator which maps the demands into *low-level directives (i.e., OF messages) to establish a chained set of network paths connecting VF end-points by enforcing consistent flow entries across a set of switches. The set of switches considered for the path set-up are firstly selected based on their current load (i.e., overloaded switches are avoided).* Then the shortest path algorithm is applied to find the final set of switches for service chain path set-up. *Moreover, the SDN orchestrator offers adaptation capabilities for the established paths (i.e., redirection of service chains paths or part of thereofs) to recover from congestion events* at switches/links or from SLA violations of service data *that are likely to occur when a concurrent resource usage takes place. To this purpose, the SDN orchestrator collects traffic statistics from the switches and elaborates them to evaluate their current load and the actual throughput performance of data flows. Firstly, t*he SDN orchestrator acts at the network *infrastructure orchestration* level by dynamically arranging service chain paths to minimize throughput degradations due to congestions with benefits in terms of balanced usage of network nodes and links. Secondly, since the regulated usage of resources does not imply that QoS requirements are strictly addressed, the SDN orchestrator also performs *resource orchestration* by detecting deviations of service data flow throughput from the SLA and by putting in place path redirections to assure QoS of established flows as required [23]. *Accordingly*, we have conceived the following two orchestration policies:

A. *Network-aware Flow Redirection (NFR): the throughput of the switches connected to cloud platforms deploying VFs is assessed. If the throughput is higher than a given threshold, the switch is considered as overloaded and the SDN orchestrator tries to redirect every active path traversing the switch throughout other available switches. The policy assures a regulated usage of switches and links and maximizes the available data throughput at the nodes.*
B. *QoS-aware Flow Redirection (QFR): the throughput of the service data flows is assessed. If the throughput value of a certain flow goes below a given threshold, the flow is redirected along another data path. This policy assures that the QoS performance are strictly preserved in line with SLA during the service chain lifecycle.*

The SDN orchestrator needs some time to perform path redirection(s) and to recover from throughput degradations or SLA deviations, i.e., redirection time. During this period, the SDN orchestrator causes *an overhead at the SDN controller*. Indeed, the redirections imply interactions with the controller for the research of alternative paths, the setup of new ones and the deletion of the overloaded ones. On the other hand, since the deletion is done after the set-up of new paths, there is not packet loss during redirections and thus no impact in terms of service data delivery.

We considered an SDN network composed of 11 switches according to the Abilene topology [22]. A subset of the switches is connected to emulated cloud platforms while the others behave as simple OpenFlow transit switches. Those switches are the ones that are monitored by the SDN orchestrator since they are more likely to be overloaded. A set of end-hosts is connected to the switches, which randomly behave as source/destination of the service chain paths. The network is emulated within the Mininet environment, which directly connects to the ONOS controller. Moreover, ONOS interacts with the proposed SDN orchestrator through a REST API to *(i)* search the shortest path between two given endpoints after passing through a predefined number of VFs, *(ii)* setup/delete the data delivery paths, *(iii)* and periodically monitor the status of the switches connected to the cloud platforms and of the installed flows. We compare the performance of the two policies with respect to a baseline where no path redirections and orchestration policies are applied. We consider the following metrics to evaluate the performance: *(i)* the average Round Trip Time (RTT) of the established flows, *(ii)* the average redirection time and *(iii)* the orchestration overhead expressed as the number of messages elaborated by the controller. Moreover, we randomly generate 100 requests according to a Poisson process. Each request asks for service chains paths connecting two end-hosts and crossing three different VFs and has an exponentially

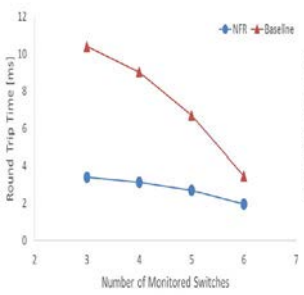

Figure 3. NFR – round trip time.

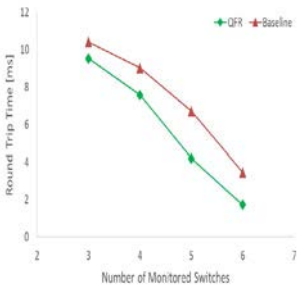

Figure 4. QFR – round trip time.

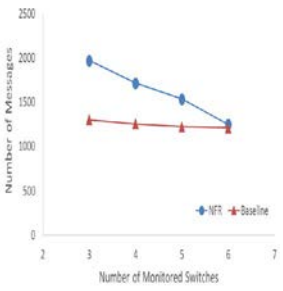

Figure 5. NFR – overhead on the controller.

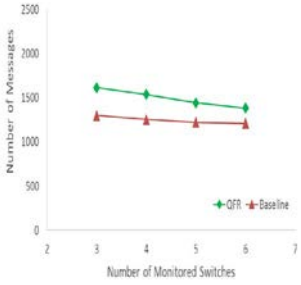

Figure 6. QFR – overhead on the controller.

distributed duration referred to as service time. The source and destination end hosts are uniformly distributed. Once an appropriate route is found, traffic is generated between the end-hosts using the *iperf* tool for the whole duration of the service time [23]. Without lack of generality, we consider that every cloud platform runs instances of three types of VFs and that it can be traversed only once by a given flow.

Figures 3 and 4 plot the RTT as a function of the number of switches connected to cloud platforms, i.e., monitored switches, for the NFR and QFR policies, respectively. Results show that an improvement is obtained in both cases since the selection of the switches is not carried out randomly as for the baseline but according to their actual status, which avoids the provision of flows traversing overloaded switches. For the NFR policy, such improvement is more relevant when the monitored switches are fewer since those switches are more loaded and thus benefit more from path redirections that are performed for all the flows traversing overloaded switches. On the other hand, the RTT is slightly higher in the QFR case since the redirections are performed for specified flows only to recover from SLA violations. However, this benefit comes to the cost of an increased overhead as shown in Fig. 5 and Fig. 6. In fact, while the overhead is almost stable for the baseline, it is higher when the redirection is applied since it implies additional interactions with the controller. More specifically, for both policies the overhead is higher when the number of switches connected to cloud platforms is low. In fact, in such a case the overall load of the switches increases (fewer options are available as for VF instances) which raises the need for path redirections. The overhead is overall higher in NFR case since the number of path redirections is generally higher than in QFR case as redirections are triggered on a per-switch basis. Finally, in Fig. 7 we plot the redirection time for both the NFR and the QFR policies. Results show that the redirection operation requires more time in the NFR case since it is performed for all the flows traversing an overloaded switch, On the contrary, in the QFR policy, the redirection involves only the flows that do not respect the agreed SLA. Moreover, the redirection time significantly decreases in the NFR case when the number of monitored switches increases because it is remarkably easier for the controller to find alternative paths for all the flows due to higher availability of VF instances and thus of alternative paths.

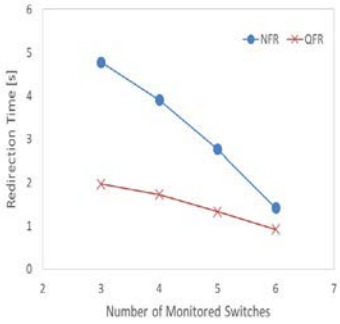

Figure 7. Redirection time: NFR vs. QFR.

4. CONCLUSIONS

A prototype of the SDN orchestrator has been presented which realizes adaptive service chain path redirections to offer different level of guarantees in service data delivery (i.e., service chain path availability and/or QoS assurance). The evaluation of the SDN orchestrator is provided in terms of average RTT and redirection time. Results show that benefits are obtained in terms of RTT at the cost of an overhead at the SDN controller that is acceptable insofar as the control channel is not congested. The redirection time is in the order of seconds, but the recovery from throughput degradations and SLA deviations is performed with no impact in terms of service data delivery to VFs.

ACKNOWLEDGEMENT

This work has been partially supported by the EU H2020 5G Exchange (5GEx) innovation project (grant no. 671636) and by EU H2020 5G-Transformer Project (grant no. 761536).

REFERENCES

[1] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, 2015.
[2] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the Internet of things realize its potential," *Computer*, vol. 49, Aug. 2016.

[3] A. Manzalini, N. Crespi, “An edge operating system enabling anything as a service,” *IEEE Communications Magazine*, Mar. 2016.
[4] F. Paganelli *et al*., “Context-aware service composition and delivery in NGSONs over SDN,” *IEEE Comm. Mag*., Aug. 2014.
[5] F. Callegati *et al*., “SDN for dynamic NFV deployment,” *IEEE Communications Magazine*, Oct. 2016.
[6] A. M. Medhat *et al*., “Service function chaining in next generation networks: State of the art and research challenges,” *IEEE Communications Magazine* , vol. PP, no. 99, pp. 2-9, Oct. 2016.
[7] Ying Zhang *et al*., “StEERING: A software-defined networking for inline service chaining,” in *Proc. 21st IEEE ICNP*, Goettingen, 2013.
[8] A. Csoma *et al*., “ESCAPE: Extensible service chain prototyping environment using Mininet, Click, Netconf and POX,” *ACM SIGCOMM Computer Commun. Rev*., vol. 44, no. 4, pp. 125-26, 2015.
[9] K. Giotis *et al*., “Policy-based orchestration of NFV services in software-defined networks,” in *Proc. NetSoft*, London, 2015.
[10] A. Manzalini *et al*, “A unifying operating platform for 5G end-2-end and multi-layer orchestration,” in *Proc. NetSoft*, Bologna, 2017.
[11] B. Martini *et al*., “A service-oriented approach for dynamic chaining of virtual network functions over multi-provider software-defined networks,” *Future Internet*, vol. 8, no. 24, 2016.
[12] B. Martini *et al*., “SDN controller for context-aware data delivery in dynamic service chaining,” in *Proc. NetSoft*, London, Apr. 2015
[13] A. A. Mohammed *et al*., “SDN controller for network-aware adaptive orchestration in dynamic service chaining,” in *Proc. NetSoft*, 2016.
[14] B. Martini *et al*., “Cross-functional resource orchestration in optical telco clouds,” in *Proc. ICTON*, Budapest, Hungary, 2015.
[15] https://thingspeak.com/
[16] www.openstack.org
[17] http://onosproject.org/
[18] I. Camelo and R. Pujar, “Intent-based VPN and its future in SDN,” in *Proc. Open Networking Summit*, USA, Mar. 2016,.
[19] L. Galluccio *et al*., “SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for wireless sensor networks,” in *Proc. INFOCOM* , May 2015.
[20] ETSI, GS NFV-MAN 001 V1.1.1, “Network functions virtualisation (NFV) management and orchestration.”
[21] “Abilene network,” https://en.wikipedia.org/wiki/Abilene Network.
[22] https://iperf.fr/
[23] M. Gharbaoui *et al*., “Network Orchestrator for QoS-enabled Service Function Chaining in reliable NFV/SDN infrastructure”, in *Proc. NetSoft*, Bologna, 2017.